

```

PPPPPPPPPPPP      AAAAAAAAAA      SSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTT      LLL
PPPPPPPPPPPPPP    AAAAAAAAAA      SSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTT      LLL
PPPPPPPPPPPPPP    AAAAAAAAAA      SSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTT      LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPPPPPPPPPPPPP    AAA          AAA          SSSSSSSSSS      RRRRRRRRRRRR      TTT              LLL
PPPPPPPPPPPPPP    AAA          AAA          SSSSSSSSSS      RRRRRRRRRRRR      TTT              LLL
PPPPPPPPPPPPPP    AAA          AAA          SSSSSSSSSS      RRRRRRRRRRRR      TTT              LLL
PPP              AAAAAAAAAAAAAAAAAA          SSS              RRR              RRR              TTT              LLL
PPP              AAAAAAAAAAAAAAAAAA          SSS              RRR              RRR              TTT              LLL
PPP              AAAAAAAAAAAAAAAAAA          SSS              RRR              RRR              TTT              LLL
PPP              AAA          AAA          SSS              RRR              RRR              TTT              LLL
PPP              AAA          AAA          SSS              RRR              RRR              TTT              LLL
PPP              AAA          AAA          SSS              RRR              RRR              TTT              LLL
PPP              AAA          AAA          SSS              RRR              RRR              TTT              LLL
PPP              AAA          AAA          SSSSSSSSSSSS      RRR              RRR              TTT              LLLLLLLLLLLLLLLLLL
PPP              AAA          AAA          SSSSSSSSSSSS      RRR              RRR              TTT              LLLLLLLLLLLLLLLLLL
PPP              AAA          AAA          SSSSSSSSSSSS      RRR              RRR              TTT              LLLLLLLLLLLLLLLLLL

```

—\$2

Sym

PAS

PAS
PASPAS
PAS

PAS

PAS

PAS
PASPAS
PAS

PAS

PAS

PAS
PAS

PAS

PAS

PAS
PASPAS
PAS

PAS

PAS

PAS
PASPAS
PAS

PAS

PAS
PAS

PAS
PAS

PAS

100

PAS

PAS

143

PAS

PAS

PAS

PAS

PAS
PAS

PAS

PAS

PAS
PASPAS
PAS

PAS

PAS

PAS
PAS

PAS

```
PPPPPPPP      AAAAAA      SSSSSSSS      RRRRRRRR      EEEEEEEEEE      AAAAAA      RRRRRRRR      EEEEEEEEEE      FFFFFFFFFF
PPPPPPPP      AAAAAA      SSSSSSSS      RRRRRRRR      EEEEEEEEEE      AAAAAA      RRRRRRRR      EEEEEEEEEE      FFFFFFFFFF
PP      PP      AA      AA      SS      RR      RR      EE      AA      AA      RR      RR      EE      FF
PP      PP      AA      AA      SS      RR      RR      EE      AA      AA      RR      RR      EE      FF
PP      PP      AA      AA      SS      RR      RR      EE      AA      AA      RR      RR      EE      FF
PP      PP      AA      AA      SS      RR      RR      EE      AA      AA      RR      RR      EE      FF
PPPPPPPP      AA      AA      SSSSSS      RRRRRRRR      EEEEEEEE      AA      AA      RRRRRRRR      EEEEEEEE      FFFFFFFF
PPPPPPPP      AA      AA      SSSSSS      RRRRRRRR      EEEEEEEE      AA      AA      RRRRRRRR      EEEEEEEE      FFFFFFFF
PP      AAAAAAAAAA      SS      RR      RR      EE      AAAAAAAAAA      RR      RR      EE      FF
PP      AAAAAAAAAA      SS      RR      RR      EE      AAAAAAAAAA      RR      RR      EE      FF
PP      AA      AA      SS      RR      RR      EE      AA      AA      RR      RR      EE      FF
PP      AA      AA      SS      RR      RR      EE      AA      AA      RR      RR      EE      FF
PP      AA      AA      SSSSSSSS      RR      RR      EEEEEEEEEE      AA      AA      RR      RR      EEEEEEEEEE      FF
PP      AA      AA      SSSSSSSS      RR      RR      EEEEEEEEEE      AA      AA      RR      RR      EEEEEEEEEE      FF
                                         ....
                                         ....
                                         ....
                                         ....
```

```
LL      IIIIII      SSSSSSSS
LL      IIIIII      SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLL      IIIIII      SSSSSSSS
LLLLLLLLLL      IIIIII      SSSSSSSS
```



```
1 0001 0 MODULE PASSREAD_REAL_F ( %TITLE 'Read an F_floating value'
2 0002 0 IDENT = '1-002' ! File: PASREAREF.B32 Edit: SBL1002
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1 *****
7 0007 1 *
8 0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
10 0010 1 * ALL RIGHTS RESERVED.
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
17 0017 1 * TRANSFERRED.
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
21 0021 1 * CORPORATION.
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
25 0025 1 *
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1
30 0030 1 ++
31 0031 1 FACILITY: Pascal Language Support
32 0032 1
33 0033 1 ABSTRACT:
34 0034 1
35 0035 1 This module contains procedures which read an F_floating value
36 0036 1 from a textfile or a string.
37 0037 1
38 0038 1 ENVIRONMENT: User mode - AST reentrant
39 0039 1
40 0040 1 AUTHOR: Steven B. Lionel, CREATION DATE: 1-April-1981
41 0041 1
42 0042 1 MODIFIED BY:
43 0043 1
44 0044 1 1-001 - Original. SBL 1-April-1981
45 0045 1 1-002 - Use PASS$END_READ. SBL 26-May-1982
46 0046 1 --
47 0047 1
```

```

: 49      0048 1 %SBTTL 'Declarations'
: 50      0049 1
: 51      0050 1 PROLOGUE DEFINITIONS:
: 52      0051 1
: 53      0052 1
: 54      0053 1 REQUIRE 'RTLIN:PASPROLOG';
: 55      0117 1 ! Externals, linkages, PSECTs, structures
: 56      0118 1
: 57      0119 1 TABLE OF CONTENTS:
: 58      0120 1
: 59      0121 1
: 60      0122 1 FORWARD ROUTINE
: 61      0123 1 PASSREAD_REAL_F;
: 62      0124 1 PASSREADV_REAL_F;
: 63      0125 1 ! Read from textfile
: 64      0126 1 ! Read from string
: 65      0127 1
: 66      0128 1 MACROS:
: 67      0129 1 NONE
: 68      0130 1
: 69      0131 1 EQUATED SYMBOLS:
: 70      0132 1
: 71      0133 1 NONE
: 72      0134 1
: 73      0135 1 FIELDS:
: 74      0136 1
: 75      0137 1 NONE
: 76      0138 1
: 77      0139 1 OWN STORAGE:
: 78      0140 1
: 79      0141 1 NONE
: 80      0142 1

```



```

82      0143 1 %SBTTL 'PASSREAD_REAL_F - Read an F_floating value from textfile'
83      0144 1 GLOBAL ROUTINE PASSREAD_REAL_F (
84      0145 1     PFV: REF $PASSPFV_FILE_VARIABLE,      ! File variable
85      0146 1     ERROR                                ! Error unwind address
86      0147 1 ) =
87      0148 1
88      0149 1 !++
89      0150 1 FUNCTIONAL DESCRIPTION:
90      0151 1
91      0152 1     This function reads an F_floating value from the specified textfile
92      0153 1     and returns it as the function value.
93      0154 1
94      0155 1 CALLING SEQUENCE:
95      0156 1
96      0157 1     Single.wf.v = PASSREAD_REAL_F (PFV.mr.r [, ERROR.ja.r])
97      0158 1
98      0159 1 FORMAL PARAMETERS:
99      0160 1
100     0161 1     PFV                - The Pascal File Variable (PFV) passed by reference.
101     0162 1     The structure of the PFV is defined in PASPFV.REQ.
102     0163 1
103     0164 1     ERROR              - Optional. If specified, the address to unwind to
104     0165 1     in case of an error.
105     0166 1
106     0167 1 IMPLICIT INPUTS:
107     0168 1
108     0169 1     NONE
109     0170 1
110     0171 1 IMPLICIT OUTPUTS:
111     0172 1
112     0173 1     NONE
113     0174 1
114     0175 1 ROUTINE VALUE:
115     0176 1
116     0177 1     The F_floating value of the number read.
117     0178 1
118     0179 1 SIDE EFFECTS:
119     0180 1
120     0181 1     If the file is the standard file INPUT or OUTPUT, it is implicitly opened.
121     0182 1
122     0183 1 SIGNALLED ERRORS:
123     0184 1
124     0185 1     INVSYNREA - invalid syntax for real value
125     0186 1     NOTVALTYP- "string" is not a value of type "type"
126     0187 1
127     0188 1 !--
128     0189 1
129     0190 2 BEGIN
130     0191 2
131     0192 2 LOCAL
132     0193 2     RESULT,                ! F_floating result value
133     0194 2     DESCR: BLOCK [8, BYTE], ! Descriptor for convert
134     0195 2     FCB: REF $PASSFCB_CONTROL_BLOCK, ! File Control block
135     0196 2     PFV_ADDR: VOLATILE,      ! Enable argument
136     0197 2     UNWIND_ACT: VOLATILE,    ! Enable argument
137     0198 2     ERROR_ADDR: VOLATILE;    ! Enable argument
138     0199 2

```

PASSREAD_REAL_F
1-002

Read an F floating value

PASSREAD_REAL_F - Read an F_floating value from

M 9
16-Sep-1984 01:59:26
14-Sep-1984 12:51:48

VAX-11 Bliss-32 V4.0-742
[PASRTL.SRC]PASREAREF.B32;1

Page 4
(3)

```

139      0200 2      BUILTIN
140      0201 2      ACTUALCOUNT;          ! Count of arguments
141      0202 2
142      0203 2      ENABLE
143      0204 2      PASS$IO_HANDLER (PFV_ADDR, UNWIND_ACT, ERROR_ADDR);      ! Enable error handler
144      0205 2
145      0206 2      !+
146      0207 2      !- Get ERROR parameter, if present.
147      0208 2
148      0209 2
149      0210 2      IF ACTUALCOUNT () GEQU 2
150      0211 2      THEN
151      0212 2      ERROR_ADDR = .ERROR;          ! Set unwind address
152      0213 2
153      0214 2      PFV_ADDR = PFV [PFV$R_PFV];      ! Set PFV address
154      0215 2
155      0216 2      !+
156      0217 2      !- Validate PFV and get PFV.
157      0218 2
158      0219 2
159      0220 2      PASS$VALIDATE_PFV (PFV [PFV$R_PFV]; FCB);
160      0221 2
161      0222 2      !+
162      0223 2      !- Set unwind action to unlock file.
163      0224 2
164      0225 2
165      0226 2      UNWIND_ACT = PASS$K_UNWIND_UNLOCK;
166      0227 2
167      0228 2      !+
168      0229 2      !- Do common initialization.
169      0230 2
170      0231 2
171      0232 2      PASS$INIT_READ (PFV [PFV$R_PFV], FCB [FCB$R_FCB]; FCB);
172      0233 2
173      0234 2      !+
174      0235 2      !- Set up string descriptor for convert call.
175      0236 2
176      0237 2
177      0238 2      DESCR [DSC$B_CLASS] = DSC$K_CLASS_S;
178      0239 2      DESCR [DSC$B_DTYPE] = DSC$K_DTYPE_T;
179      0240 2
180      0241 2      !+
181      0242 2      !- Call utility routine to find a string that looks like an real.
182      0243 2      !- If we can't find one, signal an error.
183      0244 2
184      0245 2
185      0246 2      IF NOT PASS$GET_REAL (PFV [PFV$R_PFV], FCB [FCB$R_FCB];
186      0247 2      DESCR [DSC$A_POINTER], DESCR [DSC$W_LENGTH], FCB)
187      0248 2      THEN
188      0249 2      $PASS$IO_ERROR (PASS$_INVSYNREA,2,DESCR,.FCB [FCB$L_RECORD_NUMBER]);
189      0250 2
190      0251 2      !+
191      0252 2      !- Call convert routine. If it fails, signal an error.
192      0253 2
193      0254 2
194      0255 2      IF NOT OTS$CVT_T_F (DESCR, RESULT)
195      0256 2      THEN
```



```

: 196 P 0257 2 $PASSIO ERROR (PASS NOTVALTYP,3,DESCR,
: 197 P 0258 2 UPLIT BYTE (%CHARCOUNT('SINGLE'),'SINGLE'),
: 198 0259 2 .FCB [FCB$L_RECORD_NUMBER]);
: 199 0260 2
: 200 0261 2
: 201 0262 2 !+ Do end-of-READ processing.
: 202 0263 2 !-
: 203 0264 2
: 204 0265 2 PASS$END_READ (PFV [PFV$R_PFV], FCB [FCB$R_FCB]);
: 205 0266 2
: 206 0267 2 RETURN .RESULT;
: 207 0268 2
: 208 0269 1 END;

```

! End of routine PASS\$READ_REAL_F

```

.TITLE PASS$READ_REAL_F Read an F_floating value
.IDENT \1-002\

.PSECT _PASS$CODE,NOWRT, SHR, PIC,2

45 4C 47 4E 49 06 00000 P.AAA: .BYTE 6
53 00001 .ASCII \SINGLE\

.EXTRN PASS$READ_REAL_F
.EXTRN PASS$READV_REAL_F
.EXTRN PASS$IO_HANDLER
.EXTRN PASS$VALIDATE_PFV
.EXTRN PASS$INIT_READ, PASS$GET_REAL
.EXTRN PASS$SIGNAL, PASS$INVSYNREA
.EXTRN OTSSCVT_T_F, PASS$NOTVALTYP
.EXTRN PASS$END_READ

.ENTRY PASS$READ_REAL_F, Save R2,R3,R4,R5,R6,R7,R8,-; 0144
R9
MOVAB PASS$SIGNAL, R9
SUBL2 #24, SP
CLRQ ERROR_ADDR 0190
CLRL PFV_ADDR
MOVAL 5$, (FP)
CMPB (AP), #2 0210
BLSSU 1$
MOVL ERROR, ERROR_ADDR 0212
MOVL PFV, R6 0214
MOVL R6, PFV_ADDR
JSB PASS$VALIDATE_PFV 0220
MOVL #1, UNWIND_ACT 0226
JSB PASS$INIT_READ 0232
MOVW #270, DESCR+2 0239
JSB PASS$GET_REAL 0247
MOVL R4, DESCR+4
MOVW R5, DESCR
BLBS R0, 2$
PUSHL -56(FCB) 0249
PUSHAB DESCR
PUSHL #2
MOVZBL #PASS$INVSYNREA, -(SP)
CALLS #4, PASS$SIGNAL

```

			2D	11	0005F	BRB	4\$	
			5E	DD	00061	PUSHL	SP	0255
		14	AE	9F	00063	PUSHAB	DESCR	
00000000G	00		02	FB	00066	CALLS	#2, OTSS\$CVT_T_F	
	14		50	E8	0006D	BLBS	R0, 3\$	
		C8	A7	DD	00070	PUSHL	-56(FCB)	0259
		83	AF	9F	00073	PUSHAB	P.AAA	
		18	AE	9F	00076	PUSHAB	DESCR	
			03	DD	00079	PUSHL	#3	
7E	00G		8F	9A	0007B	MOVZBL	#PASS\$K_NOTVALTYP, -(SP)	
69			05	FB	0007F	CALLS	#5, PASS\$\$SIGNAL	
			0A	11	00082	BRB	4\$	
			00	16	00084	JSB	PASS\$END_READ	0265
50	00000000G		6E	D0	0008A	MOVL	RESULT, R0	0267
				04	0008D	RET		
			50	D4	0008E	CLRL	R0	0269
				04	00090	RET		
				0000	00091	.WORD	Save nothing	0190
50		08	AC	D0	00093	MOVL	8(AP), R0	
50		04	A0	D0	00097	MOVL	4(R0), R0	
		EC	A0	9F	0009B	PUSHAB	ERROR_ADDR	
		F0	A0	9F	0009E	PUSHAB	UNWIND_ACT	
		F4	A0	9F	000A1	PUSHAB	PFV_ADDR	
			03	DD	000A4	PUSHL	#3	
			5E	DD	000A6	PUSHL	SP	
		04	AC	7D	000A8	MOVQ	4(AP), -(SP)	
00000000G	7E		03	FB	000AC	CALLS	#3, PASS\$\$IO_HANDLER	
	00			04	000B3	RET		

; Routine Size: 180 bytes,

Routine Base: _PASS\$CODE + 0007

; 209

0270 1

; 210

0271 1 !<BLF/PAGE>


```

212 0272 1 %SBTTL 'PASS$READV_REAL_F - Read an F_floating from string'
213 0273 1 GLOBAL ROUTINE PASS$READV_REAL_F (
214 0274 1     STRING: REF BLOCK [, BYTE],
215 0275 1     ERROR
216 0276 1     ) =
217 0277 1
218 0278 1 ++
219 0279 1 FUNCTIONAL DESCRIPTION:
220 0280 1
221 0281 1     This function reads an F_floating from the specified string
222 0282 1     and returns it as the function value.
223 0283 1
224 0284 1 CALLING SEQUENCE:
225 0285 1
226 0286 1     Single.wf.v = PASS$READV_REAL_F (STRING.mt.ds [, ERROR.ja.r])
227 0287 1
228 0288 1 FORMAL PARAMETERS:
229 0289 1
230 0290 1     STRING
231 0291 1     - The string to read from, passed as a class S
232 0292 1     (assumed) descriptor. The length and pointer
233 0293 1     are updated to reflect the unread string.
234 0294 1     ERROR
235 0295 1     - Optional. If specified, the address to unwind to
236 0296 1     in case of an error.
237 0297 1
238 0298 1 IMPLICIT INPUTS:
239 0299 1
240 0300 1     NONE
241 0301 1
242 0302 1 IMPLICIT OUTPUTS:
243 0303 1
244 0304 1     NONE
245 0305 1
246 0306 1 ROUTINE VALUE:
247 0307 1
248 0308 1     The value of the F_floating read.
249 0309 1
250 0310 1 SIDE EFFECTS:
251 0311 1
252 0312 1     NONE
253 0313 1
254 0314 1 SIGNALLED ERRORS:
255 0315 1
256 0316 1     NONE
257 0317 1 --
258 0318 1
259 0319 2 BEGIN
260 0320 2
261 0321 2 LOCAL
262 0322 2     PFV: $PASS$PFV_FILE_VARIABLE,
263 0323 2     RESULT,
264 0324 2     ARG_LIST: VECTOR [2, LONG],
265 0325 2     PFV_ADDR: VOLATILE,
266 0326 2     UNWIND_ACT: VOLATILE,
267 0327 2     ERROR_ADDR: VOLATILE;
268 0328 2

```



```

: 269      0329 2  BUILTIN
: 270      0330 2  ACTUALCOUNT;          ! Count of arguments
: 271      0331 2
: 272      0332 2  ENABLE
: 273      0333 2  PASS$IO_HANDLER (PFV_ADDR, UNWIND_ACT, ERROR_ADDR);    ! Enable error handler
: 274      0334 2
: 275      0335 2  !+
: 276      0336 2  ! Get ERROR parameter, if present.
: 277      0337 2  !-
: 278      0338 2
: 279      0339 2  IF ACTUALCOUNT () GEQU 2
: 280      0340 2  THEN
: 281      0341 2  ERROR_ADDR = .ERROR;          ! Set unwind address
: 282      0342 2
: 283      0343 2  PFV_ADDR = PFV [PFV$R_PFV];    ! Set PFV address
: 284      0344 2
: 285      0345 2  !+
: 286      0346 2  ! Set up ARG_LIST.
: 287      0347 2  !-
: 288      0348 2
: 289      0349 2  ARG_LIST [0] = 1;              ! One argument
: 290      0350 2  ARG_LIST [1] = PFV [PFV$R_PFV]; ! PFV address
: 291      0351 2
: 292      0352 2  !+
: 293      0353 2  ! Call PASS$DO_READV to do the work, giving it the address of
: 294      0354 2  ! PASS$READ_REAL_F to call.
: 295      0355 2  !-
: 296      0356 2
: 297      0357 2  PASS$DO_READV (PFV [PFV$R_PFV], .STRING, ARG_LIST, PASS$READ_REAL_F;
: 298      0358 2  RESULT);
: 299      0359 2
: 300      0360 2  RETURN .RESULT;
: 301      0361 2
: 302      0362 1  END;

```

! End of routine PASS\$READV_REAL_F

```

                                .EXTRN PASS$DO_READV
                                .ENTRY PASS$READV_REAL_F, Save R2,R3,R4,R6
                                SUBL2 #32, SP
                                CLRL ERROR_ADDR
                                CLRQ UNWIND_ACT
                                MOVAL 2$, (FP)
                                CMPB (AP), #2
                                BLSSU 1$
                                MOVL ERROR, ERROR_ADDR
                                MOVAB PFV, PFV_ADDR
                                MOVL #1, ARG_LIST
                                MOVAB PFV, ARG_LIST+4
                                MOVAB PASS$READ_REAL_F, R4
                                MOVAB ARG_LIST, R3
                                MOVAB PFV, R6
                                MOVL STRING, R2
                                JSB PASS$DO_READV
                                RET
                                .WORD Save nothing

```


50	08	AC	D0	00040	MOVL	8(AP), R0	:
50	04	A0	D0	00044	MOVL	4(R0), R0	:
	DC	A0	9F	00048	PUSHAB	ERROR_ADDR	:
	E0	A0	9F	0004B	PUSHAB	UNWIND_ACT	:
	E4	A0	9F	0004E	PUSHAB	PFV_ADDR	:
		03	DD	00051	PUSHL	#3	:
		5E	DD	00053	PUSHL	SP	:
00000000G	7E	04	AC	7D	MOVQ	4(AP), -(SP)	:
00		03	FB	00059	CALLS	#3, PASS\$IO_HANDLER	:
		04		00060	RET		:

; Routine Size: 97 bytes, Routine Base: _PASS\$CODE + 00BB

; 303 0363 1
; 304 0364 1 !<BLF/PAGE>

PASS\$READ_REAL_F

1-002

Read an F_floating value

PASS\$READV_REAL_F - Read an F_floating from stri

F 10

16-Sep-1984 01:59:26

VAX-11 Bliss-32 V4.0-742

14-Sep-1984 12:51:48

[PASRTL.SRC]PAS\$REAREF.B32;1

Page 10

(5)

306

0365

1

END

307

0366

1

308

0367

0

ELUDOM

! End of module PASS\$READ_REAL_F

PSECT SUMMARY

Name

Bytes

Attributes

_PASS\$CODE

284

NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

File

Total

Symbols

Percent

Pages

Processing

Loaded

Mapped

Time

\$255\$DUA28:[SYSLIB]STARLET.L32;1

9776

6

0

581

00:01.0

_\$255\$DUA28:[PASRTL.OBJ]PASLIB.L32;1

427

99

23

33

00:00.4

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS\$:PAS\$REAREF/OBJ=OBJ\$:PAS\$REAREF

MSRCS\$:PAS\$REAREF/UPDATE=(ENH\$:PAS\$REAREF

Size: 277 code + 7 data bytes

Run Time: 00:07.2

Elapsed Time: 00:31.7

Lines/CPU Min: 3049

Lexemes/CPU-Min: 12806

Memory Used: 84 pages

Compilation Complete

0296 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

